

C1 1 insertion point using a paste command. Also, the editor provides various  
2 commands (e.g., "Paste =") to insert ~~new~~ a new node at the current insertion point.

3  
4 Substitute the paragraph starting at page 13, line 32, with the following:

C2 5 In one embodiment, the IP tree editor includes a tokenizer for receiving  
6 keyboard entered text, recognizing a token, and converting the token into a  
7 sequence of editor commands. The IP tree editor provides commands for selecting  
8 portions of an IP tree and for placing an insertion point in the IP tree. The IP tree  
9 editor allows various commands to be performed relative to the currently selected  
10 portion and the current insertion point. For example, the "Paste if" command  
11 replaces the currently selected portion of the IP tree by ~~an~~ a node pointing to the  
12 declaration node that defines the IP computational construct for conditional  
13 execution, which is generally denoted as "if" in current programming language  
14 syntax. The typing of an "if" token results in the corresponding "Paste if"  
15 command being executed, relative to the current selection.

16  
17 Substitute the paragraph starting at page 16, line 1, with the following:

C3 18 Alternatively, the crown can be selected by placing the mouse pointer over  
19 the operator and ~~the~~ then alt-clicking the left mouse button.

20  
21 Substitute the paragraph starting at page 29, line 34, with the following:

C4 22 In step 804, the routine displays the representation on the display device.  
23 The routine displays an insertion point indication at a position within the display  
24 representation that corresponds to the insertion point. In a preferred embodiment,  
25

C4  
1 the routine preferably displays only a portion of long display representations, and  
2 allows the user to scroll within the display representation, causing the routine to  
3 display a different portion of the IP tree. The routine preferably generates in step  
4 803 only as much of the display representation of the IP tree as is necessary to  
5 produce the displayed portion of the IP tree.

6  
7 Substitute the paragraph starting at page 30, line 20, with the following:

C5  
8 In step 901, the routine allows the user to select an insertion point with  
9 reference to the display representation. In this step, the routine allows the  
10 programmer to select as the insertion point any position in the display  
11 representation that corresponds to either a node of the IP tree or a separation  
12 between nodes of the IP tree. The programmer may select the insertion point by  
13 pointing to the display representation using the pointing device. The programmer  
14 may also use the keyboard to select the insertion point by pressing a combination  
15 of cursor positioning keys that moves the cursor to the insertion point within the  
16 display representation. The programmer may also use an insertion point  
17 positioning command provided by the IP system. These commands move the  
18 insertion point to a given position in the IP tree, either absolute or relative to the  
19 current position of the insertion point.

20  
21 Substitute the paragraph starting at page 30, line 32, with the following:

C6  
22 In step 903, the routine allows the programmer to select the type of node  
23 that should be inserted at the insertion on point. In a preferred embodiment, the  
24 programmer uses the keyboard to enter a token corresponding to a paste command  
25

1 (e.g., "if" for the "paste if" command). Alternately, the routine displays a list of  
2 paste commands for available node types and allows the user to select one. The  
3 routine preferably specifies the types of nodes which may be inserted as children  
4 of each node type. After the user has selected to paste a node type, the routine  
5 verifies that a node of the selected type may properly be inserted at the insertion  
6 point. This involves checking that a node of this type is a proper child of the node  
7 above the insertion point. If a node of the selected type may not properly be  
8 inserted at the insertion point, the routine preferably permits the programmer to  
9 choose another node type insertion point.

10  
11 Substitute the paragraph starting at page 31, line 35, with the following:

12 The command well editor controls configuration of the user interface of the  
13 IP tree editor. The IP system provides a command well that contains each  
14 command provided by the IP system. In a preferred embodiment, each command  
15 has associated with it a unique identifier and function for implementing the  
16 command. A preferred command well is described in U.S. Patent No. 5,287,514,  
17 entitled "METHOD AND SYSTEM FOR CUSTOMIZING A USER  
18 INTERFACE IN A COMPUTER SYSTEM." The IP tree editor provides a user  
19 interface that includes a menu hierarchy, various toolbars, and various keymaps.  
20 The command well editor allows a programmer to customize the user interface.  
21 The command well editor allows a programmer to specify the menu hierarchy. A  
22 programmer can specify the names of menu items and the corresponding  
23 command to perform when the menu item is selected. The user interface  
24 preferably includes an arbitrary number of toolbars that can be individually and  
25

1 optionally placed on any of the four borders of IP's main window (oriented  
2 horizontally or vertically) according to the user's preferences. The command well  
3 editor allows a programmer to specify which commands are to be included on the  
4 toolbars. Each command typically has a bitmap specifying a button for the  
5 toolbar. The user interface preferably includes an arbitrary number of keymaps  
6 that are chosen during the course of input based on the context of the current  
7 selection, *e.g.*, a left margin keymap for when the mouse is over the left margin  
8 area of the window and a program text keymap for when the mouse is over the  
9 program text area of the window. The command well editor allows a programmer  
10 to specify the mapping of a key (or mouse event) or sequence of keys to  
11 commands. For example, a double click in the program text area may map to a  
12 command to select a node, whereas a double click in the left margin may map to a  
13 command to display a subtree. Also, if a programmer maps the letters "if" (the  
14 "if" token) to the "Paste if" command in the program text keymap, then when the  
15 programmer types "if" within the program text area the "Paste if" command is  
16 executed which causes a node to be inserted at the current insertion point.

17  
18 Substitute the paragraph starting at page 34, line 11, with the following:

19 Figures 11A-11D are sample diagrams showing the contents of a subtree  
20 and the display representation at various times during the insertion of two new  
21 nodes into the subtree. Figure 11A is a sample diagram showing the contents of  
22 the subtree and the display representation before any insertion has been performed.  
23 The subtree 1110 is defined by its root display node, a print node 1111, specifying  
24 a display operation. The print node is a child of another node, such as a grouping  
25

cg  
1 node, not shown. An "A" variable reference node 1112 and a "IS THE  
2 ANSWER" string node 1113 are children of the print node. The insertion point  
3 left selection 1114, shown as an "[ ]" symbol, is located on the line above the  
4 print node. The extent of the left selection, as indicated by the shading of the  
5 nodes, includes nodes 1111, 1112, and 1113. The display representation 1120  
6 contains a line of text 1121 corresponding to the subtree. An insertion point left  
7 selection 1122 is shown as a vertical bar cursor positioned at the beginning of the  
8 line and an ~~underseored~~ underscore indicating the extent of the selection.

9  
10 Substitute the paragraph starting at page 35, line 16, with the following:

cg  
11 The programmer then selects the "B" variable reference node type to insert  
12 at the insertion point and inserts it. The "B" variable reference node may be  
13 selected and ~~insert~~ inserted by typing it directly from the keyboard or via a copy  
14 and paste operation. Figure 11D is a sample diagram showing of the IP tree and  
15 the display representation after the IP tree editor has inserted a "B" variable  
16 reference node as a child of the multiplication node. In the subtree, the IP tree  
17 editor has inserted a "B" variable reference node 1171 to replace the placeholder  
18 node. If "B" is entered via the keyboard, then IP editor will place the insertion  
19 point to just after "B" and leave the user in interpreted selection mode in  
20 anticipation that the expression may be extended by the user. This situation is  
21 illustrated in Figure 11D by the vertical cursor bar 1172 shown just after the "B"  
22 in node 1171. In the display, item 1180, the interpreted selection insertion point is  
23 shown as a vertical cursor bar, item 1174, with no explicit extent indication.

1 Substitute the paragraph starting at page 36, line 14, with the following:

2 Figure 12C is a diagram of the display list showing the expansion of  
3 display list entry 1224. The IP system selects display list entry 1224 to expand  
4 because it is the first unexpanded display list entry in the display list. Display list  
5 entry 1224 has been expanded into display list items 1228-1230. Display list entry  
6 1228 is unexpanded, and its content indicator indicates the "A" variable reference  
7 node 1213. Display list entry 1229 is expanded, and its content indicator indicates  
8 a "\*" string 1241. Display list entry ~~1224~~ 1230 is unexpanded, and its content  
9 indicator indicates the "B" variable reference node 1214.

10 Substitute the paragraph starting at page 38, line 6, with the following:

11  
12 The IP system starts the reduction process by invoking the ProcessTE  
13 function passing the root tree element of a copy of the IP tree. The ProcessTE  
14 function loops invoking each Match function checking for a match and, when a  
15 match is found, invokes the corresponding Xform function. The Xform function  
16 typically replaces the pointer to the tree element it is passed with a pointer to a tree  
17 element that is the root of a reduced subtree and then recursively invokes the  
18 ProcessTE function to ~~processes~~ process the reduced subtree. For example, the  
19 Xform function for the \_List IP computational construct recursively invokes the  
20 ProcessTE function each tree element pointed to by an operand tree component.  
21 When an Xform function has completed the reduction of the subtree that was  
22 passed to it, it returns the resultant (transformed) subtree which will be used by the  
23 calling function to replace one of the operands in its subtree, thereby completing  
24 one of the steps in transforming its subtree.